Supplementary Material: Efficiently Creating 3D Training Data for Fine Hand Pose Estimation

Markus Oberweger Gernot Riegler Paul Wohlhart Vincent Lepetit
Institute for Computer Graphics and Vision
Graz University of Technology, Austria
{oberweger,riegler,wohlhart,lepetit}@icg.tugraz.at

Abstract

We present additional evaluation and data for our paper. In detail, we present the relaxations of the optimization problems that we actually solve. Further, we describe the autoencoder architecture, and we show an evaluation of error distributions.

1. Relaxations of Optimization Problems

We show the relaxations of the optimization problems in the paper. For all problems, we use the penalty method [5] for equality constraints. For this, we introduce a multiplier, that penalizes constraint violations.

1.1. Relaxation of Optimization Section 3.2

The given optimization problem is:

$$\begin{aligned} \underset{\{L_{r,k}\}_{k=1}^{K}}{\arg\min} & \sum_{k=1}^{K} v_{r,k} \|\mathrm{proj}(L_{r,k}) - l_{r,k}\|_2^2 \\ \text{s.t.} \quad \forall k \quad & \|L_{r,k} - L_{r,p(k)}\|_2^2 = d_{k,p(k)}^2 \\ \forall k \quad & v_{r,k} = 1 \Rightarrow \mathcal{D}_r[l_{r,k}] < z(L_{r,k}) < \mathcal{D}_r[l_{r,k}] + \epsilon \\ \forall k \quad & v_{r,k} = 1 \Rightarrow (L_{r,k} - L_{r,p(k)})^\top \cdot c_{r,k} > 0 \\ \forall k \quad & v_{r,k} = 0 \Rightarrow z(L_{r,k}) > \mathcal{D}_r[l_{r,k}] \end{aligned} \tag{1}$$

It includes an equality and inequality constraints. By introducing multipliers μ and β, the relaxation is:

$$\begin{aligned} \underset{\{L_{r,k}\}_{k=1}^{K}}{\arg\min} & \sum_{k=1}^{K} v_{r,k} \|\mathrm{proj}(L_{r,k}) - l_{r,k}\|_2^2 + \mu^b \sum_k \left(\|L_{r,k} - L_{r,p(k)}\|_2^2 - d_{k,p(k)}^2 \right)^2 + \\ & \beta \sum_{\{k | v_{r,k}=1\}} \min\left((L_{r,k} - L_{r,p(k)})^\top \cdot c_{r,k}, 0 \right)^2 \\ \text{s.t.} \quad \forall k \quad & v_{r,k} = 1 \Rightarrow \mathcal{D}_r[l_{r,k}] < z(L_{r,k}) < \mathcal{D}_r[l_{r,k}] + \epsilon \\ \forall k \quad & v_{r,k} = 0 \Rightarrow z(L_{r,k}) > \mathcal{D}_r[l_{r,k}] \end{aligned} \tag{2}$$

We start with $\mu^0 = 10^0$ and increase $\mu^{b+1} = 10\mu^b$ for 5 times. For each μ we solve the problem until convergence, *i.e.* the decrease of the objective function is smaller than 10^{-5}. Further, we keep $\beta = 1$ fixed. We use the SLSQP algorithm [2] to solve the relaxed problem, and the remaining two constraints can be integrated as box constraints in SLSQP.

1.2. Relaxation of Optimization Section 3.3

The optimization problem is:

$$\begin{aligned} &\arg\min_{\{L_{\hat{c},k}\}_k} \sum_k \mathrm{ds}(\mathcal{D}_{\hat{c}}, \mathrm{proj}(L_{\hat{c},k}); \mathcal{D}_{\hat{a}}, l_{\hat{a},k})^2 \\ &\quad \text{s.t.} \quad \forall k \quad \|L_{\hat{c},k} - L_{\hat{c},p(k)}\|_2^2 = d_{k,p(k)}^2 \ . \end{aligned} \tag{3}$$

We relax the equality constraint by introducing a multiplier γ:

$$\arg\min_{\{L_{\hat{c},k}\}_k} \sum_k \mathrm{ds}(\mathcal{D}_{\hat{c}}, \mathrm{proj}(L_{\hat{c},k}); \mathcal{D}_{\hat{a}}, l_{\hat{a},k})^2 + \gamma^b \sum_k \left(\|L_{\hat{c},k} - L_{\hat{c},p(k)}\|_2^2 - d_{k,p(k)}^2 \right)^2 \ . \tag{4}$$

We start with $\gamma^0 = 10^1$ and increase $\gamma^{b+1} = 10\gamma^b$ for 5 times. For each γ we solve the problem until convergence using the Levenberg-Marquardt [4] algorithm.

1.3. Relaxation of Optimization Section 3.4

The optimization problem is:

$$\begin{aligned} &\sum_{i \in [1;N] \setminus \mathcal{R}} \sum_k \mathrm{ds}(\mathcal{D}_i, \mathrm{proj}(L_{i,k}); \mathcal{D}_{\hat{i}}, l_{\hat{i},k})^2 + \\ &\lambda_M \sum_i \sum_k \|L_{i,k} - L_{i+1,k}\|_2^2 + \\ &\lambda_P \sum_{r \in \mathcal{R}} \sum_k v_{r,k} \|\mathrm{proj}(L_{r,k}) - l_{r,k}\|_2^2 \\ &\quad \text{s.t.} \quad \forall i,k \quad \|L_{i,k} - L_{i,p(k)}\|_2^2 = d_{k,p(k)}^2 \ . \end{aligned}$$

The equality constraint is relaxed by introducing a multiplier η:

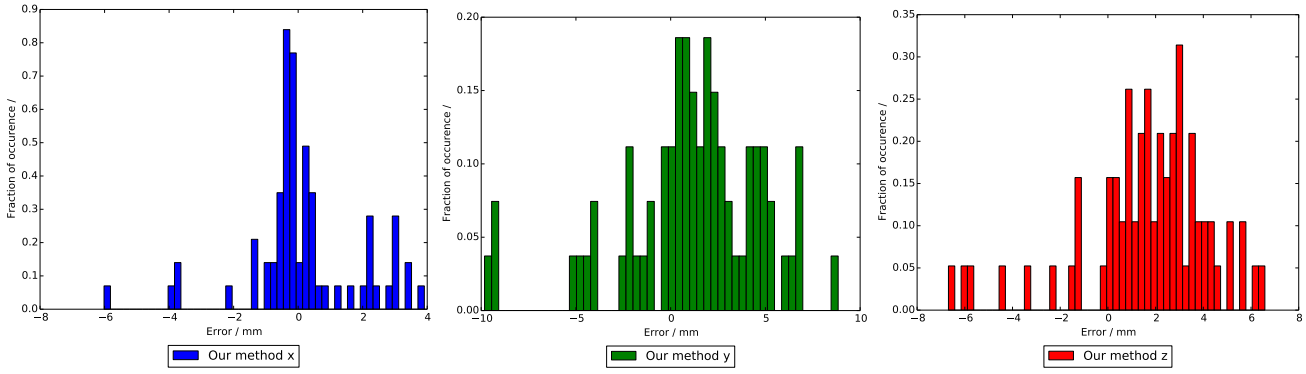
$$\begin{aligned} &\sum_{i \in [1;N] \setminus \mathcal{R}} \sum_k \mathrm{ds}(\mathcal{D}_i, \mathrm{proj}(L_{i,k}); \mathcal{D}_{\hat{i}}, l_{\hat{i},k})^2 + \\ &\lambda_M \sum_i \sum_k \|L_{i,k} - L_{i+1,k}\|_2^2 + \\ &\lambda_P \sum_{r \in \mathcal{R}} \sum_k v_{r,k} \|\mathrm{proj}(L_{r,k}) - l_{r,k}\|_2^2 + \\ &\eta^b \sum_i \sum_k \left(\|L_{i,k} - L_{i,p(k)}\|_2^2 - d_{k,p(k)}^2 \right)^2 \ . \end{aligned}$$

We start with $\eta^0 = 10^2$ and increase $\eta^{b+1} = 10\eta^b$ for 5 times. The relaxed problem is solved with sparse Levenberg-Marquardt [4] algorithm.

2. Autoencoder Architecture and Training

The encoder part of the autoencoder consists of four convolutional layers, each with 32 kernels of size 5×5. Each convolutional layer is followed by a max-pooling layer with a window size of 2×2. The convolutional layers are followed by three fully connected layers with 1024 neurons each. The decoder part is a mirrored version of the encoder, but instead of max-pooling layers, we use unpooling layers [6], which upsample the feature maps by a factor of 2. The size of the embedding is 1024 dimensions. All layers have ReLU activation functions [3].

The autoencoder is trained by minimizing the ℓ_2-norm of the reconstruction of the input image. We pre-train the autoencoder in a greedy, layer-wise fashion [1]. We train each layer for 100 epochs, using stochastic gradient descent with a momentum of 0.9 and a learning rate of 0.01 which decreases over the epochs. The batch size is 128. After pre-training, we fine-tune the full autoencoder for 100 epochs with a reduced learning rate of 0.001.



(a) (b) (c)

Figure 1: Evaluation of error distributions. We show the distance between our inferred joint locations and the ground truth joint locations for the x-, y-, and z-coordinate. (Best viewed in color)

3. Error Analysis on Synthetic Data

In Fig. 1 we show the distribution of the errors in x-, y-, z-coordinates for the synthetic data described in the paper. The errors in all coordinates are similarly distributed, thus there is no error bias towards a certain coordinate.

References

[1] G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786), 2006.

[2] D. Kraft. A Software Package for Sequential Quadratic Programming. Technical report, German Aerospace Center, 1988.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems*, 2012.

[4] K. Levenberg. A Method for the Solution of Certain Non-Linear Problems in Least Squares. *Quarterly of Applied Mathematics*, 2, 1944.

[5] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 2006.

[6] M. D. Zeiler, G. W. Taylor, and R. Fergus. Adaptive Deconvolutional Networks for Mid and High Level Feature Learning. In *Proc. of ICCV*, 2011.